Patent
Attorney's Docket No. 0120-032

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of	)
	)
Anthony SPENCER	) Confirmation No.: 4839
	)
Application No.: 10/534,343	) Group Art Unit: 2416
	)
Filed: July 18, 2005	) Examiner: Abdulla A. Riyami
	)
For: PACKET STORAGE SYSTEM FOR TRAFFIC HANDLING	)

APPEAL BRIEF PURSUANT TO 37 C.F.R § 41.37

Sir:

Further to the Notice of Appeal filed on September 16, 2009, and in connection with the above-identified application, submitted herewith is the Appeal Brief.

(i) **REAL PARTY IN INTEREST**

The real party in interest is ClearSpeed Technology Ltd.

(ii) **RELATED APPEALS AND INTERFERENCES**

To the best of the undersigned's knowledge, there are no related appeals or interferences.

(iii) **STATUS OF CLAIMS**

Claims 1-16 are currently pending, have all been rejected two or more times, and are all the subject of this appeal.

(iv) **STATUS OF AMENDMENTS**

No Amendments have been submitted in this application subsequent to the Final Official Action of April 29, 2009. A Request for Reconsideration was filed on June 20, 2009. The request was considered as indicated by the Advisory Action mailed on August 6, 2009.

(v) **SUMMARY OF CLAIMED SUBJECT MATTER**

Independent Claim 1 is directed to a method of queuing variable size data packets (see, e.g., Figure 1 and paragraph 30) in a communication system. The method includes generating from each said data packet a record portion of predetermined fixed size and containing information about the packet (see, e.g., Figure 1 and paragraph 30). The method further includes storing only data portions of said packets in independent memory locations in a first memory 2 (see, e.g., Figure 1 and paragraph 31). The method further includes storing only said record portions in one or more managed queues in a second memory 3 having fixed size memory locations equal in size to the size of the record portions (see, e.g., Figure 1 and paragraphs 30-31, and paragraph 9). The first memory 2 is larger than the second memory 3 (see, e.g., paragraph 9). The memory locations in the first memory are arranged in blocks having a plurality of different sizes and the memory locations are allocated to the data portions according to the size of the data portions (see, e.g., paragraph 9).

Independent claim 9 is directed to a memory hub for queuing received data packets (see, e.g., Figure 1 and paragraph 30). The memory hub includes an arrivals block 1, adapted to generate from each said data packet a record portion of predetermined fixed size and containing information about the packet (see, e.g., Figure 1 and paragraph 30). The memory hub further includes a first memory 2 for storing only data portions of said packets in independent memory locations (see, e.g., Figure 1 and paragraph 31). The memory hub further includes a second memory 3 for storing only said record portions in one or more managed queues, the memory having fixed size

memory locations equal in size to the size of the record portions (see, e.g., Figure 1 and paragraphs 30-31, and paragraph 9). The first memory 2 is larger than the second memory 3. The memory locations in the first memory are arranged in blocks having a plurality of different sizes and the memory locations are allocated to the data portions according to the size of the data portions (see., e.g., paragraph 9).

None of the independent claims or argued dependent claims contains an element expressed as a means or step for performing a specified function without the recital of structure, material, or acts in support thereof.

(vi) **GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL**

Appellant requests review of the following grounds of rejection on appeal.

Claims 1-4, 6-12, and 14[1]-16 were rejected under 35 U.S.C. § 103(a) as unpatentable over Mittal et al. (U.S. Patent No. 7,035,212, herein Mittal) in view of Beshai et al. (U.S. Patent Publication No. 2004/0213291, herein Beshai).

Claims 5 and 13 were rejected under 35 U.S.C. § 103(a) as unpatentable over Mittal in view of Beshai further in view of Radhakrishnan et al. (U.S. Patent Publication No. 2004/0022094, herein Radhakrishnan).

[1] According to paragraph 9 of the Final Official Action, claims 1-4, 6-12 and 13-16 stand rejected as being unpatentable under 35 U.S.C. §103 over Mittal in view of Beshai. The undersigned believes there may be an error in the claim numbering since it appears from the remainder (e.g., paragraph 10) of the Final Official Action that Claims 1-4, 6-12 and 14-16 were intended.

(vii) **ARGUMENTS**

I. Rejection of Claims 1-4, 6-12, and 14-16 is Improper

It is respectfully submitted that the record fails to establish that the proposed combination of Mittal and Beshai discloses each and every feature of independent Claims 1 and 9. Accordingly, it is respectfully submitted that the record fails to establish a prima facie case of obviousness of claims 1-4, 6-12, and 14-16.

a. The Primary Citation to Mittal Does Not Teach or Suggest Storing Only Data Portions (i.e., only the Packet Payload) in a first memory (i.e., the Ingress Memory)

Independent Claim 1 subject matter has been discussed above. Independent Claim 9, although different from Claim 1, recites the novel and non-obvious features of Claim 1 to be discussed next.

Claim 1 specifically recites

> generating from each said data packet a record portion of predetermined fixed size and containing information about the packet,;
> **storing only data portions of said packets in independent memory locations in a first memory**;
> storing only said record portions in one or more managed queues in a second memory having fixed size memory locations equal in size to the size of the record portions[.] (Emphasis added).

In other words, Claim 1 clearly presents a distinction between a data portion of a data packet and a record portion of data packet.[2] Only the data portion of a data packet is stored in one memory (2), while only the record portion of the packet is stored in another memory (3).

On the contrary, there is no teaching or suggestion in Mittal that **only** the data portion of a packet (i.e., the packet payload of Mittal) is stored in one memory. Rather, Mittal only discloses using packet headers in the control and scheduling of packets, and storing entire data packets -- including their headers -- in a memory.

The Final Official Action has erred in interpreting "only data portions" of data packets to include the entire data packet, including the data packet header. Rather, the logical interpretation of "only data portions" with respect to Mittal is "packet payload." To uphold the Final Official Action's interpretation not only completely removes all patentable weight from the expressly-recited claim term "only", but it would be in direct contradiction to Mittal's own distinction between data packets (packet headers and packet payloads) and packet payloads.

Mittal thus clearly does not disclose storing packet payloads without packet headers as discussed next.

The description in columns 1-3 relating to Figure 1A of Mittal indicate that incoming data packets from a source port 14 are allocated an ingress flow ID by the source port 14. The ingress flow ID is inserted into a field in the packet headers for

[2] It is respectfully submitted that this distinction is expressed in independent Claim 1. As such, it is respectfully submitted that the statement appearing on page 3 of the Advisory Action that "[t]here has been no distinction between the data portion and record portion in the claim" is factually incorrect and without basis.

each of the packets in the stream. A flow is defined as a packet stream with the packets in the stream having similar parameters, such as bandwidth or Class of Service (CoS) or other requirements. The packets are stored in ingress memory 20 on the basis of the flow ID in the packet header.

The ingress memory hub 18 informs the ingress traffic manager 16 of the packet ID and packet size of each packet. The ingress traffic manager sends scheduling information for that packet back to the ingress hub 18, whereby indicating when packets for different ingress flow IDs are to be output to a switch fabric 22. The ingress flow manager schedules the flows according to the ingress flow IDs and the ingress memory hub causes the next packet in the ingress memory 20 to be read out. The ingress memory hub inserts an egress flow ID into the packet header before it is sent to the switch fabric. Packets are queued in an ingress queue on a per flow basis or on a CoS basis. The reverse procedure occurs at the egress side of the switch fabric, using egress memory hub 26, memory 24 and traffic manager 28.

Column 4 of Mittal describes Figure 2, which is a detailed diagram of the ingress traffic manager and ingress memory hub of Figure 1. Packets are shown as comprising a “packet payload” 58 and “header” 56 containing the ingress flow ID. A number N of ingress queues 42 are provided in the ingress traffic manager 16, each corresponding to a respective ingress queue 46 in the memory hub 18, one per ingress flow ID. The ingress queues 42 each contain a field 43 tracking the total number of packets (in the queue) and a field 45 tracking the total length of all packets in the associated ingress

flow. Each ingress queue also includes an ordered queue 47 that identifies the length of each packet received for that particular ingress flow ID in packet arrival order.

An ingress packet arriving at the ingress memory hub 18 is written into the ingress memory 20 according to the ingress flow ID in the packet header. The egress flow ID associated with the ingress flow ID is loaded into field 48 and a forwarding label associated with that ingress flow ID is loaded into field 50, these IDs having been set up at the outset in source port 14 of Figure 1 of Mittal. When a packet is scheduled to be output, a traffic manager controller 40 informs the memory hub controller 44 to read the packet for the ingress flow ID identified by the traffic manager controller 40. The memory hub controller 44 modifies the header for then outbound packet by adding the forwarding label and egress flow ID from the respective fields 50 and 48.

It is clear from the description in Mittal that, although the packet headers are used in the control and scheduling of packets, there is absolutely no teaching or suggestion that **only** the packet payload, i.e., the data portions of the packets as per Applicant's claims, are stored in the ingress memory described by Mittal. Applicant respectfully submits that Mittal had ample opportunity to make this distinction, especially since the division of packets into header and payload portions was acknowledged in Figure 2 and various fields were discussed in relation to Figure 2, as also outlined above. The fact that Mittal did not make that distinction can only be taken by one of ordinary skill in the art as instead disclosing the ingress memory stores data packets per se, including payload **and** header. This is in stark contrast to Applicant's claims.

This being so, it is clear that Mittal potentially suffers from the same defects as other prior art packet processors, where the header and payload portions are dealt with as a single entity. Applicant expressly sets out to avoid these defects by separating out the header portions from an incoming stream of packets and passing the data portions directly to a memory where data portions of varying sizes can be allocated to comparable sized memory locations while the header portions are operated on in one or more managed queues.

The passage in Column 4, lines 31-40 referred to by the Examiner in connection with Mittal and claim 1 clearly states "[t]he memory hub controller 44 writes the **packet** 54 into ingress memory 20 according to the ingress flow ID in header 56..." (emphasis added). It is also clear from Figure 2 of Mittal at least that the packet 54 consists of the packet payload 58 **and** the packet header 56. There can be no doubt that Mittal discloses that the whole packet, i.e., payload **and** header, is stored in the ingress memory. In stark contrast to Mittal, Applicant's claim 1 recites **only** the data portions (equivalent to the payload portion in Mittal) being stored in independent memory locations in a first memory and **only** record portions (equivalent to the header in Mittal) being stored in one or more managed queues in a second memory.

Thus, for the reasons discussed above, it is respectfully submitted that Mittal clearly does not teach or suggest storing only data portions of said packets in independent memory locations in a first memory.

b. The Primary Citation to Mittal Does Not Teach or Suggest That the Claimed Second Memory Has Fixed Size Memory Locations Equal in Size To The Size Of The Record Portions

As noted above, Independent Claim 1 specifically recites

> storing only said record portions in one or more managed queues in a second memory **having fixed size memory locations equal in size to the size of the record portions**[.] (Emphasis added).

In other words, the record portions are stored in a memory (3) having fixed size locations equal in size to the record portions.

On the contrary, there is no teaching or suggestion in Mittal regarding the **size** of the locations for the record portions relative to the **size** of the record portions themselves (i.e., the header portions 56 of Mittal). Rather, Mittal only discloses that the ingress queue is "associated with" flow ID, length and CoS, and a field 43 tracking the total number of packets.

The Final Official Action has erred in interpreting "equal in size to the size of the record portions" to mean "associated with." Rather, the logical interpretation of "equal in size" with respect to Mittal is that the locations for header portions must be equal in size relative to the size of the header portions themselves. To uphold the Final Official Action's interpretation would be contrary to the term "equal".

The Final Official Action quotes column 4, lines 5-30 of Mittal for disclosing the above feature. This passage does mention a header 56 identifying an "ingress flow ID,

the length, and possibly a Class of Service (CoS) for the packet." The passage goes on to state that "[e]ach ingress queue 46 is associated with a particular ingress flow ID and contains pointers to locations in ingress memory." The passage further states that "[e]ach ingress queue 42 includes a field 43 that tracks the total number of packets and a field 45 that tracks the total length for all the packets for the associated ingress flow." Although this passage does mention an ingress queue being "associated with" flow ID, length, and Cos, and a field 43 tracking the total number of packets, these relationships bear no relation to the **size** of the memory locations relative to the **size** of the packet headers. It is respectfully submitted that it is not possible to deduce from this passage the equality of the size of the locations for the record portions relative to the size of the record portions themselves.

Thus, for the reasons discussed above, it is respectfully submitted that Mittal does not teach or suggest storing only said record portions in one or more managed queues in a second memory having fixed size memory locations equal in size to the size of the record portions.

c. The Primary Citation to Mittal Does not Teach or Suggest Allocating the Memory Locations in the First Memory from a Pool of Available Addresses Provided to It in Batches from a Central Pool of Available Addresses

Dependent Claim 4 specifically recites

> allocating the memory locations in said first memory from a pool of available addresses provided to it in batches from a central pool of available addresses.

In other words, memory locations for the data portions are allocated from a local pool 6 of addresses. The local pool 6 keeps a batch of available addresses sent from a central pool 7 of addresses.

On the contrary, there is no teaching or suggestion in Mittal regarding either a local pool of addresses or a central pool of addresses. Nor is there any teaching or suggestion regarding sending a batch of available addresses. Rather, Mittal only discloses a memory hub controller 44 writing a packet into an ingress memory 20 according to an ingress flow ID in the packet's header 56, notifying a traffic manager controller 40 of the ingress flow ID and the length of the packet, and the traffic manager controller 40 directing discard of packets for the flow ID upon the occurrence of ingress flow backup.

The Final Office Action has erred in interpreting "local pool of addresses" and "central pool of addresses" as the memory hub controller 44 and the traffic manager controller 40. Rather, the logical interpretation of "local pool of addresses" and "central pool of addresses" is two actual pools of addresses, one local and one remote. To uphold the Final Official Action's interpretation would be contrary to the term "pool." Further, to uphold the Final Official Action's interpretation would be to completely ignore the express claim language "addresses provided to [the local pool] in batches from [the] central pool of address."

The Final Official Action cites three specific passages for disclosing the above feature. Regarding the first two citations, the Final Office Action cites column 4, lines 31-40, and column 5, lines 15-20. These passages mention a memory hub controller 44 writing a packet into an ingress memory 20 according to an ingress flow ID in the packet's header 56, notifying a traffic manager controller 40 of the ingress flow ID and the length of the packet, and the traffic manager controller 40 directing discard of packets for the flow ID upon the occurrence of ingress flow backup. Nowhere in these passages is a pool of addresses so much as mentioned. Rather, what is discussed is allocating memory based on a flow ID.

Regarding the third citation, the Final Office Action cites column 7, lines 36-65. This passage discloses an example scenario in which unicast packets A, B, and C (Figure 5) are forwarded. In this example, the ingress memory hub 18 stores the packets in the ingress memory 20, and updates ingress queues #1 and #2 (corresponding to flow IDs #1 and #2). As with the previous citations, nowhere in this passage is a pool of addresses mentioned.

Moreover, despite the lack of a discussion regarding a local pool of available addresses and a central pool of available address, these passages say nothing regarding **a batch** of addresses being communicated. Claim 4 specifically recites "addresses provided to [the local pool] in batches from [the] central pool of address." As outlined above, Mittal only discusses a memory hub controller 44 notifying a traffic manger controller 40 of an ingress flow ID. To the extent the Final Official Action is

relying on the memory hub controller 44 as a local pool of available addresses and the traffic manager controller 40 as a central pool of addresses, it is respectfully submitted that the "notifying" is from the memory hub controller -- not the traffic manager controller. Further, this notification is not of a batch of addresses.

Thus, for the reasons discussed above, it is respectfully submitted that Mittal does not teach or suggest allocating the memory locations in said first memory from a pool of available addresses provided to it in batches from a central pool of available addresses.

Dependent Claim 12, although different from Claim 4, recites the novel and non-obvious feature of Claim 4 discussed above.

d. The Secondary Citation to Beshai fails to Remedy the Above-Discussed Deficiencies of Mittal

Beshai relates to "segmenting concatenated variable-size packets of a data stream." As stated therein, Beshai has as an object "develop[ing] a method of transferring variable-size packets in fixed-size data segments without incurring a significant capacity waste or unacceptable segment-formation delay."

Beshai is cited for allegedly teaching Applicant's feature that the first memory is larger than the second memory. However, it is respectfully submitted that Beshai fails to remedy the above discussed deficiencies of Mittal.

Accordingly, it is respectfully submitted that the record fails to establish that the proposed combination of Mittal and Beshai discloses each and every feature of independent Claims 1 and 9, and dependent Claims 4 and 12. Accordingly, it is respectfully submitted that the record fails to establish a prima facie case of obviousness of Claims 1-4, 6-12, and 14-16.

II. The Rejection of Claims 5 and 13 Is Improper

It is respectfully submitted that the record fails to establish that the proposed combination of Mittal, Beshai, and Radhakrishnan discloses each and every feature of independent Claims 1 and 9. Accordingly, it is respectfully submitted that the record fails to establish a prima facie case of obviousness of claims 1-4, 6-12, and 14-16.

Initially, it is noted that the U.S. filing date of Radhakrishnan was February 5, 2003, and that Applicant's priority date is November 11, 2002. Radhakrishnan does claim priority from an earlier filed provisional application. However, the Official Action has failed to even allege that the earlier filed provisional contains support for the sections of Radhakrishnan relied upon in the present rejection. Accordingly, it is respectfully submitted that the Examiner's burden to establish a prima facie case of obviousness cannot have been met, since the Examiner has not established that the evidence being relied upon in the rejection constitutes prior art.[3]

[3] Further, it is respectfully noted that the legal question of whether a 35 U.S.C. § 119(e) priority claim shifts the effective reference date of a patent to the filing date of the patent's provisional application is currently pending

Regardless, Claims 5 and 13 ultimately depend from independent Claims 1 and 9, which as discussed above, are believed to patentably distinguish over the proposed combination of Mittal and Beshai. Since Radhakrishnan does not remedy the above discussed deficiencies of Mittal and Beshai, it is respectfully submitted that no combination of Mittal, Beshai, and Radhakrishnan could have motivated one of ordinary skill in the art to have arrived at Applicant's claim 5 and 13 combinations.

Conclusion

As the Examiner did not show that all claimed elements are taught or inherent in the applied art, reversal of all outstanding rejections is respectfully requested.

Respectfully submitted,
POTOMAC PATENT GROUP PLLC

By: /stevenmdubois/
Steven M. duBois
Registration No. 35,023

Dated: November 16, 2009

Customer No. 42015
Potomac Patent Group PLLC
P.O. Box 270
Fredericksburg, VA 22404
(540) 361-1863

---

before the U.S. Court of Appeals for the Federal Circuit. In re Giacomini et al., Appeal No. 2009-1400, Application Serial No. 09/725,737.

(viii) **CLAIMS APPENDIX**

1. A method of queuing variable size data packets in a communication system, the method comprising:

generating from each said data packet a record portion of predetermined fixed size and containing information about the packet,;

storing only data portions of said packets in independent memory locations in a first memory;

storing only said record portions in one or more managed queues in a second memory having fixed size memory locations equal in size to the size of the record portions;

wherein:

the first memory is larger than the second memory; and

the memory locations in the first memory are arranged in blocks having a plurality of different sizes and the memory locations are allocated to the data portions according to the size of the data portions.

2. A method as claimed in claim 1, wherein there are two sizes of memory location in the first memory arranged in two said blocks, one of a size to receive data portions of a first size and the other of a size to receive data portions of a second size, the second size being larger than the first size, and wherein data portions that are too large to be stored in a single memory block are stored as linked lists in a plurality of blocks with pointers pointing to the next block.

3. A method as claimed in claim 1, wherein the sizes of the memory locations in the blocks are matched to the most commonly occurring sizes of data packets in the communication system.

4. A method as claimed in claim 1, further comprising allocating the memory locations in said first memory from a pool of available addresses provided to it in batches from a central pool of available addresses.

5. A method as claimed in claim 4, wherein the memory blocks are segregated into a plurality of memory channels, the method further comprising allocating addresses to data portions sequentially across channels whereby to spread the storage of data portions across the channels.

6. A method as claimed in claim 4, further comprising reading the data portions from the first memory in pipelined manner by a data retrieval unit adapted to instruct a memory block to read out a data portion without having to wait for the previous read to be completed, and releasing the address location from the first memory.

7. A method as claimed in claim 1, further comprising, under circumstances where there is insufficient memory for a received packet, enqueueing the record portion as though the corresponding data portion was stored in the first memory, subsequently reading out the record portion corresponding to the said data packet, setting a flag to indicate that the data portion of the said packet is to be discarded, discarding the said data portion, and releasing the memory location notionally allocated to the discarded data portion.

8. A method as claimed in claim 6, further comprising reading the address locations from a bitmap of addresses and, when a memory location is released after the data stored therein has been read out, the address of the released memory location is sent directly to the pool.

9. A memory hub for queuing received data packets, comprising:

an arrivals block, adapted to generate from each said data packet a record portion of predetermined fixed size and containing information about the packet;

a first memory for storing only data portions of said packets in independent memory locations;

a second memory for storing only said record portions in one or more managed queues, the memory having fixed size memory locations equal in size to the size of the record portions;

wherein:

the first memory is larger than the second memory; and

the memory locations in the first memory are arranged in blocks having a plurality of different sizes and the memory locations are allocated to the data portions according to the size of the data portions.

10. A memory hub as claimed in claim 9, wherein there are two sizes of memory location in the first memory arranged in two said blocks, one of a size to receive data portions of a first size and the other of a size to receive data portions of a second size, the second size being larger than the first size, and wherein data portions that are too large to be stored in a single memory block are stored as linked lists in a plurality of blocks with pointers pointing to the next block but without any pointers pointing from one data portion to the next data portion of the packet.

11. A memory hub as claimed in claim 9, wherein the sizes of the memory locations in the blocks are matched to the most commonly occurring sizes of data packets in the communication system.

12. A memory hub as claimed in claim 9, wherein the memory locations in said first memory are allocated from a pool of available addresses provided to it in batches from a central pool of available addresses.

13. A memory hub as claimed in claim 12, wherein the memory blocks are segregated into a plurality of memory channels, and addresses are allocated to data portions sequentially across channels whereby to spread the storage of data portions across the channels.

14. A memory hub as claimed in claim 12, further comprising a data retrieval unit adapted to read out the data portions from the first memory in pipelined manner and to instruct a memory block to read out a data portion without having to wait for the previous read to be completed, and releasing the address location from the first memory.

15. A memory hub as claimed in claim 9, further comprising flag setting means such that, under circumstances where there is insufficient memory for a received packet, the record portion is enqueued as though the corresponding data portion was stored in the first memory, the record portion corresponding to the said data packet is subsequently read out, and the flag setting means sets a flag to cause the data portion of the said packet to be discarded and the memory location notionally allocated to the discarded data portion released.

16. A memory hub as claimed in claim 14, further comprising a bitmap of address locations and means operable such that, when a memory location is released after the data stored therein has been read out, the address of the released memory location is sent directly to the pool.

(ix) **EVIDENCE APPENDIX**

None.

(x) **RELATED PROCEEDINGS APPENDIX**

None.